Detailed Failure Report for our test runs

Test number	Test Purpose	How output failed	Code error	How it was fixed
All tests	All testing purposes	All the tests contain errors or failed at first, because we have implemented the transaction summary statement in the wrong format.	Error was in code. We forgot the part which contains the to-account-number in most transactions	By re-coding the transaction statement part, we have overcome the issue
All tests	All testing purposes	All the tests contain errors or failed at first, because of a small typo in terms of logical issue in our *loginstat* variable.	Error was in code. Code contains a formatting typo. Instead of "==" to compare boolean values, we used "=" instead	We change the statement in which we compare two booleans values into "==" and the issue was fixed
All tests	All testing purposes	All the tests contain errors or failed because of the code "EOS" was not handled correctly	Error was in code. transactionCheck does not have a if statement when the transaction statement is "EOS"	By adding the if part into transaction check
tsR7T1	Check the unused account number part in a transaction statement - should be filled with zero	Instead of giving expected output, which is true, it gives the error in which the account number starts with 0	Error was in code. We missed unused constraints when implementing the constraints that account number cannot begin with zero	By adding multiple conditions in the function transactionCheckCons4
wiR1T3	Check depositAmount	Test case failed because the monetary amount is not displaying right.	Error was in test. I didn't write the monetary amount in cents but rather in dollars when implementing the test case.	I have modified the test code's monetary amount from 1000 to 100000 to represent it in cents instead of in dollars

wiR1T5b	Check if account has enough to withdraw	Test case failed, without displaying error prompt “insufficient funds”, because we didn’t implement this feature in the code.	Error was in code. We forgot to implement this feature in the code.	We have added and re-implemented this feature in our code. So now when there’s no sufficient balance, the withdrawal would fail with the error prompt.
deR1T11	Check functionality for depositing 2000 in Agent	Test case failed, because the depositing functionality is not working as expected due to coding formatting issues in our test code.	Error was in test. I didn’t write the deposit account number (i.e. 1234567) followed by the monetary amount in cents (i.e. 200000) as the formatting convention, as the fourth terminal input. Rather I splitted them as two separate terminal input.	We have changed the test code a bit by having the fourth terminal input as deposit account number followed by the monetary amount in cents. (i.e. 1234567200000)
crR2T1	Check if new account number entered are all digits	Test case failed, because the expected output is “Invalid account number” but the actual output is “Not separated by space”	Error was in code. We noticed that there’s always a space existing at the beginning of the content, which is possibly the reason why causes this sort of failure output,	We have re-implemented the code to the point where the Scanner and Split would ignore or eliminate the redundant space in the content.
deR1T10	Check functionality for depositing 1500 in Agent	Test case failed, because the expected output is “Deposited” but the actual output is “Exceeding length”	Error was in code. We missed unused constraints when implementing the constraints where the length is exceeding when it’s greater than 61.	We have re-implemented and added multiple conditions in the class of deposit in order to meet this constraint.
crR3T1	check if the new account number is exactly the same	Test case failed, because the expected output is “Account	Error was in code. We forgot and missed to implement this	Fixed this issue by re-implementing and adding multiple

	as the currently in-use account numbers	number used” but the actual output is “invalid account”	feature in our code base.	conditions in the class of accounts in order to meet this feature
DeR1T5	Check super big number	Test case failed, because the expected output is “invalid input” but the actual output is “out of limit”	Error was in code. We have messed up to actually differentiate between an invalid input and a deposit daily monetary limit	Fixed the problem by re-implementing and added multiple if-else conditions in the class of deposit in order to detect a super big number as out-of-limit input rather than invalid input
trR1T2	Transfer $1000000.00 Out-of-boundary test: Confirms that the max amount to transfer is $999,999.99 in agent mode	Test case failed, because the actual output is “out of limit” but the wrongly written expected output is “Invalid output”	Error was in test. I wrote the expected output in the test code incorrectly by writing it “Invalid output” rather than “out of limit”.	We fixed the problem by changing this test case’s expected output from “Invalid output” to “out of limit” in the test code.
All tests	All testing purposes	All the tests contain errors or failed at first, because we have implemented the test code with a small typo in our logical condition.	Error was in test. All the testing input account length should be 7, but when implementing the test cases, we wrote it wrong by writing != 7	We fixed the problem by changing the test cases’ account length’s condition from “!= 7” to “==7”. And this made a lot of errors passed.
All tests	All testing purposes	The biggest issue we have encountered is write to file appending each time for the subsequent tests that runs.	Error was in code. The issue occurs within the code where the file appending part is being dealt with.	We fixed it by initiating all variables in the main function before calling other functions.
loR3T1	illegal mode input: mode names other than "agent" and "machine" are	We supposed that it will output "EOS" "EOS" because the designation of the test is not to permit a	Error was in test. The test’s expected output is written incorrectly.	We fixed the problem by changing this test case’s expected output from "EOS" "EOS" to "EOS" in

	illegal	terminal input of “login logout transfer” and that “EOS” may indeed appear twice.		the test code.
trR2T1	Within-boundary test: Confirms that it does correctly transfer the right amount to the right account (for machine ATM)	Test case failed, because, given a balance of 1 million, the expected output is written incorrectly as we expected that it may transfer successfully but instead it should output “EOS”	Error was in test. The test’s expected output is written incorrectly.	We fixed the problem by changing this test case’s expected output from "Transferred" to "EOS" in the test code.